J 6

```
NNN         NNN  MMM         MMM  LLL
NNN         NNN  MMM         MMM  LLL
NNN         NNN  MMM         MMM  LLL
NNN         NNN  MMMMMM   MMMMMM  LLL
NNN         NNN  MMMMMM   MMMMMM  LLL
NNN         NNN  MMMMMM   MMMMMM  LLL
NNNNNN      NNN  MMM   MMM   MMM  LLL
NNNNNN      NNN  MMM   MMM   MMM  LLL
NNNNNN      NNN  MMM   MMM   MMM  LLL
NNN   NNN   NNN  MMM         MMM  LLL
NNN   NNN   NNN  MMM         MMM  LLL
NNN   NNN   NNN  MMM         MMM  LLL
NNN      NNNNNN  MMM         MMM  LLL
NNN      NNNNNN  MMM         MMM  LLL
NNN      NNNNNN  MMM         MMM  LLL
NNN         NNN  MMM         MMM  LLL
NNN         NNN  MMM         MMM  LLL
NNN         NNN  MMM         MMM  LLL
NNN         NNN  MMM         MMM  LLLLLLLLLLLLLLL
NNN         NNN  MMM         MMM  LLLLLLLLLLLLLLL
NNN         NNN  MMM         MMM  LLLLLLLLLLLLLLL
```

```
_$
Ps
NP
NP
$G
$O
NP
PA
_L
```

```
**FILE**ID**NMLPARINI                                          C  4
```

```
NN      NN  MM      MM  LL          PPPPPPPP      AAAAAA    RRRRRRRR      IIIIII    NN      NN    IIIIII
NN      NN  MM      MM  LL          PPPPPPPP      AAAAAA    RRRRRRRR      IIIIII    NN      NN    IIIIII
NN      NN  MMMM  MMMM  LL          PP      PP  AA      AA  RR      RR      II      NN      NN      II
NN      NN  MMMM  MMMM  LL          PP      PP  AA      AA  RR      RR      II      NN      NN      II
NNNN    NN  MM  MM  MM  LL          PP      PP  AA      AA  RR      RR      II      NNNN    NN      II
NNNN    NN  MM  MM  MM  LL          PP      PP  AA      AA  RR      RR      II      NNNN    NN      II
NN  NN  NN  MM      MM  LL          PPPPPPPP    AA      AA  RRRRRRRR        II      NN  NN  NN      II
NN  NN  NN  MM      MM  LL          PPPPPPPP    AA      AA  RRRRRRRR        II      NN  NN  NN      II
NN    NNNN  MM      MM  LL          PP          AAAAAAAAAA  RR  RR          II      NN    NNNN      II
NN    NNNN  MM      MM  LL          PP          AAAAAAAAAA  RR  RR          II      NN    NNNN      II
NN      NN  MM      MM  LL          PP          AA      AA  RR    RR        II      NN      NN      II          ....
NN      NN  MM      MM  LL          PP          AA      AA  RR    RR        II      NN      NN      II          ....
NN      NN  MM      MM  LLLLLLLLLL  PP          AA      AA  RR      RR    IIIIII    NN      NN    IIIIII        ....
NN      NN  MM      MM  LLLLLLLLLL  PP          AA      AA  RR      RR    IIIIII    NN      NN    IIIIII        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0001  0 %TITLE 'NML initial message parsing module'
0002  0 MODULE NML$PARINI (
0003  0                 LANGUAGE (BLISS32),
0004  0                 ADDRESSING_MODE (NONEXTERNAL=GENERAL),
0005  0                 ADDRESSING_MODE (EXTERNAL=GENERAL),
0006  0                 IDENT = 'V04-000'
0007  0                 ) =
0008  1 BEGIN
0009  1 !
0010  1 !****************************************************************************
0011  1 !*                                                                          *
0012  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0013  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0014  1 !*  ALL RIGHTS RESERVED.                                                    *
0015  1 !*                                                                          *
0016  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0017  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0018  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0019  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0020  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0021  1 !*  TRANSFERRED.                                                            *
0022  1 !*                                                                          *
0023  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0024  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0025  1 !*  CORPORATION.                                                            *
0026  1 !*                                                                          *
0027  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0028  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0029  1 !*                                                                          *
0030  1 !*                                                                          *
0031  1 !****************************************************************************
0032  1 !
0033  1
0034  1 !++
0035  1 ! FACILITY:  DECnet-VAX V2.0 Network Management Listener
0036  1 !
0037  1 ! ABSTRACT:
0038  1 !
0039  1 !       This module contains action routines called by NPARSE to process
0040  1 !       NICE command messages from NCP.
0041  1 !
0042  1 ! ENVIRONMENT:  VAX/VMS Operating System
0043  1 !
0044  1 ! AUTHOR:  Distributed Systems Software Engineering
0045  1 !
0046  1 ! CREATION DATE:  8-OCT-1979
0047  1 !
0048  1 ! MODIFIED BY:
0049  1 !
0050  1 !       V03-012 MKP0012         Kathy Perko             23-July-1984
0051  1 !               If area 0 is supplied in a node number, default to the
0052  1 !               executor node area number.  This undoes the change dated
0053  1 !               21-Mar-1984.
0054  1 !
0055  1 !       V03-011 MKP0011         Kathy Perko             18-April-1984
0056  1 !               Get the executor ID from the volatile database on an as
0057  1 !               needed basis, but only once per command (rather than reissuing
1 !                     the QIO every time the exec ID is needed.)  Do it once per
1 !                     command in case the command changes the name or address.
1 !
1 !     V03-010 MKP0010         Kathy Perko             21-Mar-1984
1 !             Add support for area 1 problem.  This involves changing area 0
1 !             to area 1 for Phase IV NCPs and to the exec area for Phase III
1 !             NCPs.  Also, disallow anything but SHOW and LIST from a Phase
1 !             III node.  If they try to do a SET NODE by node number, they'll
1 !             get area 1 instead of the exec's area - very confusing.
1 !
1 !     V03-009 MKP0009         Kathy Perko             6-Jan-1984
1 !             Add X25-Access Module entity.
1 !
1 !     V03-008 MKP0008         Kathy Perko             4-Aug-1983
1 !             Add support to make node permanent database faster.
1 !
1 !     V03-007 MKP0007         Kathy Perko             20-April-1983
1 !             Remove service functions from NML.
1 !
1 !     V03-006 MKP0006         Kathy Perko             17-Jan-1983
1 !             Add support for CONFIGURATOR module.
1 !
1 !     V03-005 MKP0005         Kathy Perko             14-Nov-1982
1 !             Add a routine to return success if the NICE message
1 !             function code is change.
1 !
1 !     V03-004 MKP0004         Kathy Perko             8-Nov-1982
1 !             Change NML$PRSID so that it will save a field using the
1 !             field length in the parsing tables.
1 !
1 !     V03-003 MKP0003         Kathy Perko             15-Oct-1982
1 !             Change the way NML$PRSID saves node numbers, logging
1 !             sinks, and link numbers so that they are a longword instead
1 !             of a word.
1 !
1 !     V03-002 MKP0002         Kathy Perko             17-June-1982
1 !             Add support for active X25-protocol networks.
1 !             Also, add a routine for parsing quaifiers and
1 !             change LINKS operations to use the node number or
1 !             name as a qualifier.
1 !
1 !     V03-001 MKP0001         Kathy Perko             16-June-1982
1 !             Add parsing routines for X25-Protocol Module and entity
1 !             qualfiers.
1 !
1 !     V02-003 MKP0002         Kathy Perko             23-Nov-1981
1 !             Delete NML validation of line and circuit IDs.  NETACP
1 !             will perform all validation.
1 !
1 !     V02-002 MKP0001         Kathy Perko             13-Nov-1981
1 !             Change name of routine that used to parse line ids
1 !             and now parses both line and circuit ids.  I.E. change
1 !             NML$PRSLINE to NML$PRSDEVICE.
1 !
1 !     V02-001 LMK0001         Len Kawell              27-Jul-1981
1 !             Remove QIO buffer initialization.
1 !--
```

```
  1 %SBTTL 'Declarations';
  1
  1 !
  1 ! TABLE OF CONTENTS:
  1 !
  1
  1 FORWARD ROUTINE
  1     nml$parse_init,
  1     nml$prsfnc,
  1     nml$prsopt,
  1     nml$prsop2,
  1     nml$prsinf,
  1     nml$prsent,
  1     nml$prsidleq,
  1     nml$prsqualleq,
  1     nml$prsid,
  1     nml$prsidn,
  1     nml$prsnodnam,
  1     nml$prs_node_num_entity,
  1     nml$prs_node_num,
  1     nml$prssnknna,
  1     nml$prssnknad,
  1     nml$prs_module,
  1     nml$prs_active_net,
  1     nml$prsexesnk,
  1     nml$prsdevice,
  1     nml$prslogsin,
  1     nml$prs_noread,
  1     nml$prserr1,
  1     nml$prsiderr;
  1
  1 !
  1 ! INCLUDE FILES:
  1 !
  1
  1 LIBRARY 'LIB$:NMLLIB.L32';
  1 LIBRARY 'SHRLIB$:NMALIBRY.L32';
  1 LIBRARY 'SYS$LIBRARY:STARLET.L32';
  1
  1 !
  1 ! MACROS:
  1 !
  1
  1 !
  1 ! Macro to return a byte complement of a value
  1 ! (Used to prevent byte initialization overflow)
  1 !
  1 MACRO
M 1     not_byte (n) =
M 1         ((NOT (n)) AND %X'FF')
  1     %;
  1
  1 !
  1 ! EQUATED SYMBOLS:
  1 !
  1
  1 LITERAL
```

```
	funcnt = 7;			! Total number of functions (Phase III only)
!
! Invalid option bit mask definitions
!
LITERAL
	rea_invob_msk = not_byte (nma$m_opt_ent OR
				  nma$m_opt_inf OR
				  nma$m_opt_per),

	cha_invob_msk = not_byte (nma$m_opt_ent OR
				  nma$m_opt_inf OR
				  nma$m_opt_per OR
				  nma$m_opt_cle),

	zer_invob_msk = not_byte (nma$m_opt_ent OR nma$m_opt_rea),

	loa_invob_msk = not_byte (nma$m_opt_ent),

	dum_invob_msk = not_byte (nma$m_opt_ent),

	tri_invob_msk = not_byte (nma$m_opt_ent),

	tes_invob_msk = not_byte (nma$m_opt_ent OR nma$m_opt_acc);

!
! OWN STORAGE:
!

!
! Table of invalid option bits for each function
!
BIND
	invopb_tab = UPLIT BYTE(
				loa_invob_msk,
				dum_invob_msk,
				tri_invob_msk,
				tes_invob_msk,
				cha_invob_msk,
				rea_invob_msk,
				zer_invob_msk
			       ) : VECTOR [funcnt, BYTE];

!
! EXTERNAL REFERENCES:
!

$NML_EXTDEF;

EXTERNAL
	nml$ab_npa_blk : $NPA_BLKDEF,
	nml$gb_ncp_version: BBLOCK,
	nml$gw_perm_exec_addr: WORD,
	nml$gw_vol_exec_addr: WORD,
	nml$gq_perm_exec_name_dsc: VECTOR,
	nml$gq_vol_exec_name_dsc: VECTOR,
	nml$npa_init;
```

```
1 EXTERNAL ROUTINE
1       nma$nparse,
1       nml$chkexe,
1       nml$error_1,
1       nml$error_2,
1       nml$fix_node_num,
1       nml$getexeadr,
1       nml$getexenam,
1       nml$getnodadr,
1       nml$openfile,
1       nml$set_up_exec_id;
```

```
1 %SBTTL 'NML$PARSE_INIT  Initial message parsing routine'
1 GLOBAL ROUTINE NML$PARSE_INIT =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine invokes the NPARSE facility to check the funcition,
1 !       option, and entity codes in a received NICE protocol function.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NML$GB_FUNCTION contains the function code.
1 !       NML$GB_OPTIONS contains the option codes.
1 !       NML$GB_INFO contains the information code if the function is read.
1 !       NML$GL_ENTCODE contains the entity code.
1 !       NML$AB_NPA_BLK contains parsing information about the remainder of the
1 !               message.
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       If the parse fails then the NML status code is returned as specified in
1 !       the parse state table otherwise NML$_STS_SUC is returned.
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2 BEGIN
2
2 LOCAL
2     STATUS;                               ! Temporary status
2 !
2 ! Initialize message parsing data
2 !
2 nml$gl_prmcode   = 0;            ! Parameter code
2 nml$gl_prs_flgs  = 0;            ! Parsing flags
2 nml$gw_prmdescnt = 0;            ! Parameter descriptor count
2 nml$gl_nml_entity = 0;           ! NML's internal code for the entity.
2 nml$gw_vol_exec_addr = 0;        ! Get executor name and address from volatile
2 nml$gq_vol_exec_name_dsc [0] = 0;! db at most once for each NICE command.
2 nml$gw_perm_exec_addr = 0;       ! Get executor name and address from perm
2 nml$gq_perm_exec_name_dsc [0] = 0;! db at most once for each NICE command.
2 !
2 ! Call the NPARSE facility to parse function, option, and entity
2 !
2 nml$ab_npa_blk [npa$l_msgptr] = nml$ab_rcvbuffer;  ! Add buffer address and
```

```
2 nml$ab_npa_blk [npa$l_msgcnt] = .nml$gl_rcvdatlen; !  length NPARSE arguments
2
2 status = nma$nparse (nml$ab_npa_blk,
2                         nml$npa_init); ! Use Phase III state table
2 RETURN .status
2
1 END;                                   ! End of NML$PARSE_INIT

                                                        .TITLE  NML$PARINI NML initial message parsing module
                                                        .IDENT  \V04-000\

                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

            78  08  08  78  F8  F8  F8  00000 P.AAA:    .BYTE   -8, -8, -8, 120, 8, 8, 120                  ;

                                          INVOPB_TAB=           P.AAA
                                                        .EXTRN  NML$GB_EVTSRCTYP
                                                        .EXTRN  NML$GQ_EVTSRCDSC
                                                        .EXTRN  NML$GW_EVTCLASS
                                                        .EXTRN  NML$GB_EVTMSKTYP
                                                        .EXTRN  NML$GQ_EVTMSKDSC
                                                        .EXTRN  NML$GW_EVTSNKADR
                                                        .EXTRN  NML$GW_ACP_CHAN
                                                        .EXTRN  NML$GL_LOGMASK, NML$GQ_ENTSTRDSC
                                                        .EXTRN  NML$AB_QIOBUFFER
                                                        .EXTRN  NML$GQ_QIOBFDSC
                                                        .EXTRN  NML$AB_EXEBUFFER
                                                        .EXTRN  NML$GL_EXEDATPTR
                                                        .EXTRN  NML$GQ_EXEDATDSC
                                                        .EXTRN  NML$GQ_EXEBFDSC
                                                        .EXTRN  NML$AB_RCVBUFFER
                                                        .EXTRN  NML$GQ_RCVBFDSC
                                                        .EXTRN  NML$AB_SNDBUFFER
                                                        .EXTRN  NML$GQ_SNDBFDSC
                                                        .EXTRN  NML$GL_RCVDATLEN
                                                        .EXTRN  NML$AB_CPTABLE, NML$AB_MSGBLOCK
                                                        .EXTRN  NML$AB_ENTITY_ID
                                                        .EXTRN  NML$AB_QUALIFIER_ID
                                                        .EXTRN  NML$AB_ENTITYDATA
                                                        .EXTRN  NML$AB_NML_NMV, NML$AB_PRMSEM
                                                        .EXTRN  NML$AB_RECBUF, NML$AL_ENTINFTAB
                                                        .EXTRN  NML$AL_PERMINFTAB
                                                        .EXTRN  NML$AW_PRM_DES, NML$GB_CMD_VER
                                                        .EXTRN  NML$GB_ENTITY_CODE
                                                        .EXTRN  NML$GB_ENTITY_FORMAT
                                                        .EXTRN  NML$GL_QUALIFIER_PST
                                                        .EXTRN  NML$GB_QUALIFIER_FORMAT
                                                        .EXTRN  NML$GB_FUNCTION
                                                        .EXTRN  NML$GB_INFO, NML$GB_OPTIONS
                                                        .EXTRN  NML$GL_PRMCODE, NML$GL_PRS_FLGS
                                                        .EXTRN  NML$GL_NML_ENTITY
                                                        .EXTRN  NML$GQ_NETRAMDSC
                                                        .EXTRN  NML$GQ_RECBFDSC
                                                        .EXTRN  NML$GW_PRMDESCNT
                                                        .EXTRN  NML$AB_NPA_BLK, NML$GB_NCP_VERSION
                                                        .EXTRN  NML$GW_PERM_EXEC_ADDR
                                                        .EXTRN  NML$GW_VOL_EXEC_ADDR
                                                        .EXTRN  NML$GQ_PERM_EXEC_NAME_DSC
                                                        .EXTRN  NML$GQ_VOL_EXEC_NAME_DSC
                                                        .EXTRN  NML$NPA_INIT, NMA$NPARSE
                                                        .EXTRN  NML$CHKEXE, NML$ERROR_1
                                                        .EXTRN  NML$ERROR_2, NML$FIX_NODE_NUM
                                                        .EXTRN  NML$GETEXEADR, NML$GETEXENAM
                                                        .EXTRN  NML$GETNODADR, NML$OPENFILE
                                                        .EXTRN  NML$SET_UP_EXEC_ID

                                                        .PSECT  $CODE$,NOWRT,2

                                       0004 00000       .ENTRY  NML$PARSE_INIT, Save R2                       ; 0241
                  52 00000000G    00     9E 00002       MOVAB   NML$AB_NPA_BLK+8, R2
                     00000000G    00     D4 00009       CLRL    NML$GL_PRMCODE                                ; 0285
                     00000000G    00     D4 0000F       CLRL    NML$GL_PRS_FLGS                               ; 0286
                     00000000G    00     B4 00015       CLRW    NML$GW_PRMDESCNT                              ; 0287
                     00000000G    00     D4 0001B       CLRL    NML$GL_NML_ENTITY                             ; 0288
                     00000000G    00     B4 00021       CLRW    NML$GW_VOL_EXEC_ADDR                          ; 0289
                     00000000G    00     D4 00027       CLRL    NML$GQ_VOL_EXEC_NAME_DSC                      ; 0290
                     00000000G    00     B4 0002D       CLRW    NML$GW_PERM_EXEC_ADDR                         ; 0291
                     00000000G    00     D4 00033       CLRL    NML$GQ_PERM_EXEC_NAME_DSC                     ; 0292
                  62 00000000G    00     9E 00039       MOVAB   NML$AB_RCVBUFFER, NML$AB_NPA_BLK+8            ; 0296
          FC      A2 00000000G    00     D0 00040       MOVL    NML$GL_RCVDATLEN, NML$AB_NPA_BLK+4            ; 0297
                     00000000G    00     9F 00048       PUSHAB  NML$NPA_INIT                                  ; 0299
                            F8    A2     9F 0004E       PUSHAB  NML$AB_NPA_BLK                                ;
     00000000G    00                  02 FB 00051       CALLS   #2, NMA$NPARSE                                ;
                                      04 00058          RET                                                   ; 0303

; Routine Size:  89 bytes,     Routine Base:  $CODE$ + 0000
;   307         0304  1 %SBTTL 'NMLSPRSFNC  Store function code (action routine)'
;   308         0305  1 GLOBAL ROUTINE NMLSPRSFNC =
;   309         0306  1
;   310         0307  1 !++
;   311         0308  1 ! FUNCTIONAL DESCRIPTION:
;   312         0309  1 !
;   313         0310  1 !       Parse and store the function code from the NICE command message.
;   314         0311  1 !
;   315         0312  1 ! FORMAL PARAMETERS:
;   316         0313  1 !
;   317         0314  1 !       NONE
;   318         0315  1 !
;   319         0316  1 ! IMPLICIT INPUTS:
;   320         0317  1 !
;   321         0318  1 !       NONE
;   322         0319  1 !
;   323         0320  1 ! IMPLICIT OUTPUTS:
;   324         0321  1 !
;   325         0322  1 !       NML$GB_FUNCTION contains the function code.
;   326         0323  1 !
;   327         0324  1 ! ROUTINE VALUE:
;   328         0325  1 ! COMPLETION CODES:
;   329         0326  1 !       If Phase III NCP and not a read function, returns NML$_STS_FUN.
;   330         0327  1 !       Otherwise, returns success (NML$_STS_SUC)
;   331         0328  1 !
;   332         0329  1 ! SIDE EFFECTS:
;   333         0330  1 !
;   334         0331  1 !       NONE
;   335         0332  1 !
;   336         0333  1 !--
;   337         0334  1
;   338         0335  2 BEGIN
;   339         0336  2
;   340         0337  2 $NPA_ARGDEF;                      ! Define NPARSE block reference
;   341         0338  2
;   342         0339  2 nml$gb_function = .nparse_block [npa$b_byte]; ! Set function
;   343         0340  2 RETURN nml$_sts_suc
;   344         0341  2
;   345         0342  1 END;                              ! End of NMLSPRSFNC


                                       0000 00000         .ENTRY  NMLSPRSFNC, Save nothing                        ; 0305
               00000000G  00      18  AC  90 00002        MOVB    24(NPARSE_BLOCK), NML$GB_FUNCTION               ; 0339
                          50          01  D0 0000A        MOVL    #1, R0                                          ; 0340
                                          04 0000D        RET                                                     ; 0342

; Routine Size:  14 bytes,    Routine Base:  $CODE$ + 0059
```

```
1 %SBTTL 'NML$PRSOPT  Check and store option byte (action routine)'
1 GLOBAL ROUTINE NML$PRSOPT =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Parse and store the options byte from the NICE command message.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NML$GB_OPTIONS contains the option byte.
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2 BEGIN
2
2 $NPA_ARGDEF;                          ! Define NPARSE block reference
2
2 LOCAL
2     invbits   : BYTE,                 ! Invalid option bit temporary
2     tab_index : SIGNED BYTE,          ! Invalid bit mask table index
2     addr,
2     status;
2 !
2 ! Check NICE message options
2 !
2 nml$gb_options = .nparse_block [npa$b_byte]; ! Save entire option byte
2 tab_index = .nml$gb_function;    ! Get function code for table index
2 tab_index = .tab_index - 15;     ! Normalize the table index
2
3 IF (.tab_index GEQ 0)
2    AND (.tab_index LSS funcnt) THEN     ! Range check
3     BEGIN
3     invbits = .invopb_tab [.tab_index] AND .nml$gb_options; ! Mask
3     IF .invbits EQLU 0 THEN
3         status = nml$_sts_suc           ! No invalid bits
3     ELSE
3         status = nml$_sts_fun           ! Unrecognized option
3     END
2 ELSE
```

```
;  404        0400  2       status = nml$_sts_mpr;                  ! State table error
;  405        0401  2   !
;  406        0402  2   ! Most NCP commands need the executor node's address and/or name at some
;  407        0403  2   ! point.  Therefore, get it now, and set up globals containing either the
;  408        0404  2   ! volatile or the permanent database executor address.
;  409        0405  2   !
;  410        0406  2   IF .status THEN
;  411        0407  2       nml$set_up_exec_id (addr);
;  412        0408  2   RETURN .status;
;  413        0409  1   END;                              ! End of NMLSPRSOPT



                                         000C 00000          .ENTRY  NMLSPRSOPT, Save R2,R3                        ; 0344
                     53 00000000G   00    9E 00002          MOVAB   NML$GB_OPTIONS, R3                            ;
                     5E             04    C2 00009          SUBL2   #4, SP                                        ;
                     63         18  AC    90 0000C          MOVB    24(NPARSE_BLOCK), NML$GB_OPTIONS              ; 0386
                     50 00000000G   00    90 00010          MOVB    NML$GB_FUNCTION, TAB_INDEX                    ; 0387
                     50             0F    82 00017          SUBB2   #15, TAB_INDEX                                ; 0388
                     50             50    98 0001A          CVTBL   TAB_INDEX, R0                                 ; 0390
                                    1D    19 0001D          BLSS    2$                                            ;
                     07             50    91 0001F          CMPB    R0, #7                                        ; 0391
                                    18    18 00022          BGEQ    2$                                            ;
                     52             63    92 00024          MCOMB   NML$GB_OPTIONS, R2                            ; 0393
           51 00000000'0040         52    8B 00027          BICB3   R2, INVOPB_TAB[R0], INVBITS                   ;
                                    05    12 00030          BNEQ    1$                                            ; 0394
                     52             01    D0 00032          MOVL    #1, STATUS                                    ; 0395
                                    08    11 00035          BRB     3$                                            ;
                     52             02    CE 00037 1$:      MNEGL   #2, STATUS                                    ; 0397
                                    03    11 0003A          BRB     3$                                            ; 0394
                     52             0A    CE 0003C 2$:      MNEGL   #10, STATUS                                   ; 0400
                     09             52    E9 0003F 3$:      BLBC    STATUS, 4$                                    ; 0406
                                    5E    DD 00042          PUSHL   SP                                            ; 0407
              00000000G   00        01    FB 00044          CALLS   #1, NML$SET_UP_EXEC_ID                        ;
                     50             52    D0 0004B 4$:      MOVL    STATUS, R0                                    ; 0408
                                          04 0004E          RET                                                   ; 0409

; Routine Size:  79 bytes,    Routine Base:  $CODE$ + 0067
;    415            0410   1 %SBTTL 'NML$PRSOP2  Store Phase II option code (action routine)'
;    416            0411   1 GLOBAL ROUTINE NML$PRSOP2 =
;    417            0412   1
;    418            0413   1 !++
;    419            0414   1 ! FUNCTIONAL DESCRIPTION:
;    420            0415   1 !
;    421            0416   1 !       Parse and store the options byte from the Phase II NICE command
;    422            0417   1 !       message.
;    423            0418   1 !
;    424            0419   1 ! FORMAL PARAMETERS:
;    425            0420   1 !
;    426            0421   1 !       NONE
;    427            0422   1 !
;    428            0423   1 ! IMPLICIT INPUTS:
;    429            0424   1 !
;    430            0425   1 !       NONE
;    431            0426   1 !
;    432            0427   1 ! IMPLICIT OUTPUTS:
;    433            0428   1 !
;    434            0429   1 !       NML$GB_OPTIONS contains the option byte.
;    435            0430   1 !
;    436            0431   1 ! ROUTINE VALUE:
;    437            0432   1 ! COMPLETION CODES:
;    438            0433   1 !
;    439            0434   1 !       Always returns success (NML$_STS_SUC).
;    440            0435   1 !
;    441            0436   1 ! SIDE EFFECTS:
;    442            0437   1 !
;    443            0438   1 !       NONE
;    444            0439   1 !
;    445            0440   1 !--
;    446            0441   1
;    447            0442   2 BEGIN
;    448            0443   2
;    449            0444   2 $NPA_ARGDEF;                       ! Define NPARSE block reference
;    450            0445   2 !
;    451            0446   2 ! Save Phase II NICE message option code
;    452            0447   2 !
;    453            0448   2 nml$gb_options = .nparse_block [npa$b_byte];
;    454            0449   2
;    455            0450   2 RETURN nml$_sts_suc
;    456            0451   2
;    457            0452   1 END;                               ! End of NML$PRSOP2




                                         0000 00000         .ENTRY  NML$PRSOP2, Save nothing                          ; 0411
                 00000000G  00       18  AC  90 00002         MOVB    24(NPARSE_BLOCK), NML$GB_OPTIONS                  ; 0448
                            50           01  D0 0000A         MOVL    #1, R0                                            ; 0450
                                             04 0000D         RET                                                       ; 0452

; Routine Size:  14 bytes,    Routine Base:  $CODE$ + 00B6
```

```
:   459          0453   1 %SBTTL 'NML$PRSINF  Store information type code (action routine)'
:   460          0454   1 GLOBAL ROUTINE NML$PRSINF =
:   461          0455   1
:   462          0456   1 !++
:   463          0457   1 ! FUNCTIONAL DESCRIPTION:
:   464          0458   1 !
:   465          0459   1 !       This routine is a NPARSE action routine that sets the
:   466          0460   1 !       information code if the function is read information.
:   467          0461   1 !
:   468          0462   1 ! FORMAL PARAMETERS:
:   469          0463   1 !
:   470          0464   1 !       NONE
:   471          0465   1 !
:   472          0466   1 ! IMPLICIT INPUTS:
:   473          0467   1 !
:   474          0468   1 !       NPARSE_BLOCK [NPA$B_BYTE] contains the information code.
:   475          0469   1 !
:   476          0470   1 ! IMPLICIT OUTPUTS:
:   477          0471   1 !
:   478          0472   1 !       NML$GB_INFO contains the information type code.
:   479          0473   1 !
:   480          0474   1 ! ROUTINE VALUE:
:   481          0475   1 ! COMPLETION CODES:
:   482          0476   1 !
:   483          0477   1 !       Success (NML$_STS_SUC) is always returned.
:   484          0478   1 !
:   485          0479   1 ! SIDE EFFECTS:
:   486          0480   1 !
:   487          0481   1 !       NONE
:   488          0482   1 !
:   489          0483   1 !--
:   490          0484   1
:   491          0485   2 BEGIN
:   492          0486   2
:   493          0487   2 $NPA_ARGDEF;                        ! Define NPARSE block reference
:   494          0488   2 !
:   495          0489   2 ! Save the information code from the NPARSE argument block
:   496          0490   2 !
:   497          0491   2 nml$gb_info = .nparse_block [npa$b_byte];
:   498          0492   2
:   499          0493   2 RETURN nml$_sts_suc
:   500          0494   2
:   501          0495   1 END;                                ! End of NML$PRSINF



                                          0000 00000         .ENTRY  NML$PRSINF, Save nothing                          ; 0454
                00000000G  00      18  AC  90 00002         MOVB    24(NPARSE_BLOCK), NML$GB_INFO                      ; 0491
                           50      01  D0 0000A             MOVL    #1, R0                                             ; 0493
                                           04 0000D         RET                                                        ; 0495

; Routine Size:  14 bytes,    Routine Base:  $CODE$ + 00C4
;  503        0496  1 %SBTTL 'NML$PRSENT  Store entity type code (action routine)'
;  504        0497  1 GLOBAL ROUTINE NML$PRSENT =
;  505        0498  1
;  506        0499  1 !++
;  507        0500  1 ! FUNCTIONAL DESCRIPTION:
;  508        0501  1 !
;  509        0502  1 !       This routine is a NPARSE action routine that sets the
;  510        0503  1 !       enitity code.
;  511        0504  1 !
;  512        0505  1 ! FORMAL PARAMETERS:
;  513        0506  1 !
;  514        0507  1 !       NONE
;  515        0508  1 !
;  516        0509  1 ! IMPLICIT INPUTS:
;  517        0510  1 !
;  518        0511  1 !       NPARSE_BLOCK [NPA$B_BYTE] contains the entity code.
;  519        0512  1 !
;  520        0513  1 ! IMPLICIT OUTPUTS:
;  521        0514  1 !
;  522        0515  1 !       NML$GB_ENTITY_CODE contains the entity code.
;  523        0516  1 !
;  524        0517  1 ! ROUTINE VALUE:
;  525        0518  1 ! COMPLETION CODES:
;  526        0519  1 !
;  527        0520  1 !       Success (NML$_STS_SUC) is always returned.
;  528        0521  1 !
;  529        0522  1 ! SIDE EFFECTS:
;  530        0523  1 !
;  531        0524  1 !       NONE
;  532        0525  1 !
;  533        0526  1 !--
;  534        0527  1
;  535        0528  2 BEGIN
;  536        0529  2
;  537        0530  2 $NPA_ARGDEF;                      ! Define NPARSE block reference
;  538        0531  2 !
;  539        0532  2 ! Save the entity code from the NPARSE argument block
;  540        0533  2 !
;  541        0534  2 nml$gb_entity_code = .nparse_block [npa$b_byte];
;  542        0535  2 RETURN nml$_sts_suc
;  543        0536  2
;  544        0537  1 END;                              ! End of NML$PRSENT



                                        0000 00000          .ENTRY  NML$PRSENT, Save nothing                      : 0497
                 00000000G  00      18  AC  90 00002          MOVB    24(NPARSE_BLOCK), NML$GB_ENTITY_CODE          : 0534
                            50          01  D0 0000A          MOVL    #1, R0                                        : 0535
                                            04 0000D          RET                                                   : 0537

; Routine Size:  14 bytes,    Routine Base:  $CODE$ + 00D2


;  545        0538  1
```

```
1 %SBTTL 'NML$PRSIDLEQ  Store entity format code if plural entity'
1 GLOBAL ROUTINE NML$PRSIDLEQ =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This is an action routine called while parsing a NICE command.  It
1 !       saves the entity format code if it is plural (KNOWN, ACTIVE, ADJACENT,
1 !       etc.)
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NPARSE_BLOCK [NPA$L_FLDPTR] points to the entity format code.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       The main entity format code is saved in NML$GB_ENTITY_FORMAT.
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       Success (NML$_STS_SUC) is returned if code specifies a plural
1 !       entity.  If the entity format byte specifies a single entity,
1 !       unrecognized component error (NML$_STS_CMP) is returned.
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NPARSE state table transition is rejected if error is returned.
1 !
1 !--
1
2 BEGIN
2
2 $NPA_ARGDEF;                      ! Define NPARSE block reference
2
2 LOCAL
2     temp : SIGNED BYTE;           ! Temporary format code storage
2
2 temp = .(.nparse_block [npa$l_fldptr])<0,8>; ! Get entity format code
2
2 !
2 ! If the entity format byte is less than zero, then the NICE
2 ! command specifies a plural entity.
2 !
2 IF .temp LEQ 0 THEN
3     BEGIN
3     nml$gb_entity_format = .temp;              ! Save format code
3     RETURN nml$_sts_suc
3     END
2 ELSE
2     RETURN nml$_sts_cmp           ! Return "single entity" completion.
2
1 END;                              ! End of NML$PRSIDLEQ
```

```
                                          0000 00000         .ENTRY  NMLSPRSIDLEQ, Save nothing                    ; 0540
                          50        14  BC  90 00002         MOVB    @20(NPARSE_BLOCK), TEMP                       ; 0577
                                        0B  14 00006         BGTR    1$                                            ; 0583
                   00000000G  00        50  90 00008         MOVB    TEMP, NMLSGB_ENTITY_FORMAT                    ; 0585
                              50        01  D0 0000F         MOVL    #1, R0                                        ; 0589
                                            04 00012         RET                                                   ;
                          50            10  CE 00013 1$:     MNEGL   #16, R0                                       ;
                                            04 00016         RET                                                   ; 0591

; Routine Size:  23 bytes,    Routine Base:  $CODE$ + 00E0


;  600          0592  1
```

```
1 %SBTTL 'NML$PRSQUALLEQ  Store entity format code if plural entity'
1 GLOBAL ROUTINE NML$PRSQUALLEQ =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This is an action routine called while parsing a NICE command with
1 !       an entity qualifier.  It saves the qualifier's format code if it
1 !       is plural (KNOWN, ACTIVE, ADJACENT, etc.)
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NPARSE_BLOCK [NPA$L_FLDPTR] points to the qualifier format code.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       The qualifier format code is saved in NML$GB_QUALIFIER_FORMAT.
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       Success (NML$_STS_SUC) is returned if code specifies a plural
1 !       qualifier.  If the qualifier format byte specifies a single entity,
1 !       unrecognized component error (NML$_STS_CMP) is returned.
1 !
1 ! SIDE EFFECTS:
1 !       NPARSE state table transition is rejected if error is returned.
1 !
1 !--
1
2 BEGIN
2
2 $NPA_ARGDEF;                          ! Define NPARSE block reference
2
2 LOCAL
2     temp : SIGNED BYTE;               ! Temporary format code storage
2
2 temp = .(.nparse_block [npa$l_fldptr])<0,8>; ! Get entity format code
2
2 !
2 ! If the qualifier format byte is less than zero, then the NICE
2 ! command specifies a plural entity.  Note that a KNOWN qualifier
2 ! is the same thing as no qualifier at all.
2 !
2 IF .temp LEQ 0 THEN
3     BEGIN
3     nml$gb_qualifier_format = .temp;    ! Save format code
3     RETURN nml$_sts_suc;
3     END
2 ELSE
2     RETURN nml$_sts_cmp;                ! Return "single entity" completion.
2
1 END;                                  ! End of NML$PRSQUALLEQ
```

```
                                           0000 00000          .ENTRY  NMLSPRSQUALLEQ, Save nothing                   ; 0594
                         50        14   BC   90 00002          MOVB    a20(NPARSE_BLOCK), TEMP                        ; 0630
                                        0B   14 00006          BGTR    1$                                             ; 0637
              00000000G  00             50   90 00008          MOVB    TEMP, NMLSGB_QUALIFIER_FORMAT                  ; 0639
                         50             01   D0 0000F          MOVL    #1, R0                                         ; 0643
                                             04 00012          RET                                                    ;
                         50        10   CE      00013 1$:      MNEGL   #16, R0                                        ;
                                             04 00016          RET                                                    ; 0645

; Routine Size:  23 bytes,    Routine Base:  $CODE$ + 00F7


;  655          0646  1
```

```
1 %SBTTL 'NML$PRSID  Store entity format code and id (action routine)'
1 GLOBAL ROUTINE NML$PRSID =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This is a NPARSE action routine that stores the entity format code
1 !       a specified number of bytes of entity id or qualifier id.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NPARSE_BLOCK [NPA$L_FLDPTR] points to entity format and id.
1 !       NPARSE_BLOCK [NPA$L_FLDCNT] contains length.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NML$GB_ENTITY_FORMAT contains the entity format code.
1 !       NML$AB_ENTITY_ID contains the entity id string.
1 !                       or
1 !       NML$GB_QUALIFIER_FORMAT contains the entity qualifier's format code.
1 !       NML$AB_QUALIFIER_ID contains the entity qualifier's id string.
1 !
1 !--
1
2 BEGIN
2
2 $NPA_ARGDEF;                          ! Define NPARSE block reference
2
2 LOCAL
2     count : SIGNED,
2     cpt_index,
2     cpt_entry : REF BBLOCK,
2     iptr,
2     optr;
2
2 count = .nparse_block [npa$l_fldcnt] - 1;        ! Get field count less format code
2 iptr = .nparse_block [npa$l_fldptr];    ! Get input field pointer
2
2 !
2 ! If parsing a qualifier, save the format and compute the address of the
2 ! Parameter Semantic Table (PST) entry for the qualifier (the CPT index
2 ! for the parameter is put in the NPARSE block parameter by the parsing
2 ! tables).
2 !
2 IF .nml$gl_prs_flgs [nml$v_prs_qualifier] THEN
3     BEGIN
3     optr = nml$ab_qualifier_id;
3     nml$gb_qualifier_format = CH$RCHAR_A (iptr);        ! Store format code
3     cpt_index = .nparse_block [npa$l_param];
3     cpt_entry = nml$ab_cptable [.cpt_index, 0, 0, 0, 0];
3     nml$gl_qualifier_pst =
3               nml$ab_prmsem [.cpt_entry [cpt$w_pstindex], 0, 0, 0, 0];
3     END
2 ELSE
3     BEGIN
3     optr = nml$ab_entity_id;                    ! Get pointer to entity storage
3     nml$gb_entity_format = CH$RCHAR_A (iptr);   ! Store format code
```

```
;  714          0704  2        END;
;  715          0705  2
;  716          0706  2  IF .count GTR 0 THEN
;  717          0707  2      CH$COPY (.count, .iptr, 0, 4, .optr);          ! Move entity ID, making it
;  718          0708  2                                                     !          a longword.
;  719          0709  2  RETURN nml$_sts_suc
;  720          0710  2
;  721          0711  1  END;                                   ! End of NML$PRSID



                                                    003C 00000          .ENTRY  NML$PRSID, Save R2,R3,R4,R5                    ; 0648
                         53       10    AC           01   C3 00002          SUBL3   #1, 16(NPARSE_BLOCK), COUNT                   ; 0682
                                        52       14   AC   D0 00007          MOVL    20(NPARSE_BLOCK), IPTR                         ; 0683
                         31 00000000G   00           02   E1 0000B          BBC     #2, NML$GL_PRS_FLGS, 1$                        ; 0691
                                        51 00000000G 00   9E 00013          MOVAB   NML$AB_QUALIFIER_ID, OPTR                      ; 0693
                            00000000G   00           82   90 0001A          MOVB    (IPTR)+, NML$GB_QUALIFIER_FORMAT               ; 0694
                                        50       20   AC   D0 00021          MOVL    32(NPARSE_BLOCK), CPT_INDEX                    ; 0695
                                        50           0A   C4 00025          MULL2   #10, R0                                        ; 0696
                                        50 00000000G0040  9E 00028          MOVAB   NML$AB_CPTABLE[R0], CPT_ENTRY                  ;
                                        50           60   3C 00030          MOVZWL  (CPT_ENTRY), R0                                ; 0698
                                        50           10   C4 00033          MULL2   #16, R0                                        ;
                            00000000G   00 00000000G0040  9E 00036          MOVAB   NML$AB_PRMSEM[R0], NML$GL_QUALIFIER_PST        ;
                                                     0E   11 00042          BRB     2$                                             ; 0691
                                        51 00000000G 00   9E 00044  1$:     MOVAB   NML$AB_ENTITY_ID, OPTR                         ; 0702
                            00000000G   00           82   90 0004B          MOVB    (IPTR)+, NML$GB_ENTITY_FORMAT                  ; 0703
                                                     53   D5 00052  2$:     TSTL    COUNT                                          ; 0706
                                                     06   15 00054          BLEQ    3$                                             ;
           04            00             62           53   2C 00056          MOVC5   COUNT, (IPTR), #0, #4, (OPTR)                  ; 0707
                                                     61      0005B                                                                 ;
                                        50           01   D0 0005C  3$:     MOVL    #1, R0                                         ; 0709
                                                          04 0005F          RET                                                    ; 0711

; Routine Size:  96 bytes,    Routine Base:  $CODE$ + 010E


;  722          0712  1
```

```
1 %SBTTL 'NML$PRSIDN  Store singular entity length and name (action routine)'
1 GLOBAL ROUTINE NML$PRSIDN =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This is an action routine called while parsing a NICE command if the
1 !       command specifies a singular entity (e.g. LINE DMC-0).  It saves
1 !       the entity length (in entity format code field) and the number of
1 !       bytes of entity id (up to 16).
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NPARSE_BLOCK [NPA$L_FLDPTR] contains the pointer to the entity
1 !           format code and id string.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NML$GB_ENTITY_FORMAT contains the entity format code.
1 !       NML$AB_ENTITY_ID contains the entity id string.
1 !           or
1 !       NML$GB_QUALIFIER_FORMAT contains the entity qualifier's length.
1 !       NML$AB_QUALIFIER_ID contains the entity qualifier's id string.
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       NML$_STS_SUC
1 !
1 !--
1
2 BEGIN
2
2 $NPA_ARGDEF;                          ! Define NPARSE block reference
2
2 LOCAL
2     cpt_index,
2     cpt_entry : REF BBLOCK,
2     iptr,
2     optr,
2     length;
2
2 iptr = .nparse_block [npa$l_fldptr];    ! Get input field pointer
2 length = ch$rchar_a (iptr);              ! Save entity length
2
2 ! Some NICE commands specify qualifiers to the entity.  Save the qualifier
2 ! format separately from the main entity's.  Also, use the NPARSE block
2 ! parameter, which was set to the parameter's CPT index by the parsing
2 ! table, to compute the parameter's Parameter Semantic Table (PST) entry
2 ! address.
2
2 IF .nml$gl_prs_flgs [nml$v_prs_qualifier] THEN
3     BEGIN
3     nml$gb_qualifier_format = .length;
3     optr = nml$ab_qualifier_id;
3     cpt_index = .nparse_block [npa$l_param];
3     cpt_entry = nml$ab_cptable [.cpt_index, 0, 0, 0, 0];
```

```
:   781         0770 3      nml$gl_qualifier_pst =
:   782         0771 3                  nml$ab_prmsem [.cpt_entry [cpt$w_pstindex], 0, 0, 0, 0];
:   783         0772 3      END
:   784         0773 2  ELSE
:   785         0774 3      BEGIN
:   786         0775 3      nml$gb_entity_format = .length;      ! Save format code
:   787         0776 3      optr = nml$ab_entity_id;             ! Get entity id storage pointer
:   788         0777 2      END;
:   789         0778 2  CH$MOVE (.length,
:   790         0779 2          .iptr,
:   791         0780 2          .optr);                          ! Move entity id
:   792         0781 2
:   793         0782 2  RETURN nml$_sts_suc
:   794         0783 2
:   795         0784 1  END;                                 ! End of NML$PRSIDN



                                                      003C 00000          .ENTRY   NML$PRSIDN, Save R2,R3,R4,R5                          ; 0714
                          53          14         AC   D0 00002          MOVL     20(NPARSE_BLOCK), IPTR                                ; 0755
                          51                     83   9A 00006          MOVZBL   (IPTR)+, LENGTH                                       ; 0756
              31 00000000G 00                    02   E1 00009          BBC      #2, NML$GL_PRS_FLGS, 1$                               ; 0764
                 00000000G 00                    51   90 00011          MOVB     LENGTH, NML$GB_QUALIFIER_FORMAT                       ; 0766
                          52 00000000G           00   9E 00018          MOVAB    NML$AB_QUALIFIER_ID, OPTR                             ; 0767
                          50          20         AC   D0 0001F          MOVL     32(NPARSE_BLOCK), CPT_INDEX                           ; 0768
                          50                     0A   C4 00023          MULL2    #10, R0                                               ; 0769
                          50 00000000G0040            9E 00026          MOVAB    NML$AB_CPTABLE[R0], CPT_ENTRY                         ;
                          50                     60   3C 0002E          MOVZWL   (CPT_ENTRY), R0                                       ; 0771
                          50                     10   C4 00031          MULL2    #16, R0                                               ;
                 00000000G 00 00000000G0040            9E 00034          MOVAB    NML$AB_PRMSEM[R0], NML$GL_QUALIFIER_PST               ;
                                                 0E   11 00040          BRB      2$                                                    ; 0764
                 00000000G 00                    51   90 00042 1$:      MOVB     LENGTH, NML$GB_ENTITY_FORMAT                          ; 0775
                          52 00000000G           00   9E 00049          MOVAB    NML$AB_ENTITY_ID, OPTR                                ; 0776
              62          63                     51   28 00050 2$:      MOVC3    LENGTH, (IPTR), (OPTR)                                ; 0780
                          50                     01   D0 00054          MOVL     #1, R0                                                ; 0782
                                                      04 00057          RET                                                            ; 0784

; Routine Size:  88 bytes,    Routine Base:  $CODE$ + 016E
```

```
1 %SBTTL 'NML$PRSNODNAM  Check node name against executor (action routine)'
1 GLOBAL ROUTINE NML$PRSNODNAM =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This is a NPARSE action that checks the node name against the
1 !       the name of the executor node name.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NPARSE_BLOCK [NPA$L_FLDPTR] contains the pointer to the entity
1 !           format code and id string.
1 !       NML$GL_PRS_FLGS contains the current message parsing flag information.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NML$GB_ENTITY_FORMAT contains the entity format code.
1 !       NML$AB_ENTITY_ID contains the entity id string.
1 !       NML$GL_NML_ENTITY is set to NML$C_EXECUTOR if this is the executor
1 !       node.
1 !
1 !--
1
2 BEGIN
2
2 $NPA_ARGDEF;                            ! Define NPARSE block reference
2
2 BUILTIN
2     CALLG;
2
2 MAP
2     nml$gb_options : BBLOCK [1];
2
2 LOCAL
2     namptr,
2     namlen,
2     exenambuf : VECTOR [6, BYTE],
2     exenamdsc : DESCRIPTOR,
2     exenamlen,
2     status;
2
2 exenamdsc [dsc$w_length] = 6;
2 exenamdsc [dsc$a_pointer] = exenambuf;
2
2 namptr = .nparse_block [npa$l_fldptr] + 1;
2 namlen = .nparse_block [npa$l_fldcnt] - 1;
2 !
2 ! If the node name in the NICE command matches the executor node name
2 ! then set the internal NML entity type to executor.
2 !
2 IF nml$chkexe (nma$c_pcno_nna, 0, .namlen, .namptr) THEN
2     nml$gl_nml_entity = nml$c_executor;
```

```
;  854          0842  2 !
;  855          0843  2 ! Parse the node id normally.
;  856          0844  2 !
;  857          0845  2 CALLG (.nparse_block, nml$prsidn);
;  858          0846  2 RETURN nml$_sts_suc
;  859          0847  2
;  860          0848  1 END;                                   ! End of nml$prsnodnam



                                        0000 00000        .ENTRY  NML$PRSNODNAM, Save nothing                  ; 0786
                        5E                 10  C2 00002        SUBL2   #16, SP                                      ;
                        6E                 06  B0 00005        MOVW    #6, EXENAMDSC                                ; 0831
                 04     AE         08      AE  9E 00008        MOVAB   EXENAMBUF, EXENAMDSC+4                       ; 0832
            51   14     AC                 01  C1 0000D        ADDL3   #1, 20(NPARSE_BLOCK), NAMPTR                 ; 0834
            50   10     AC                 01  C3 00012        SUBL3   #1, 16(NPARSE_BLOCK), NAMLEN                 ; 0835
                                           03  BB 00017        PUSHR   #^M<R0,R1>                                   ; 0840
                                           7E  D4 00019        CLRL    -(SP)                                        ;
                        7E        01F4     8F  3C 0001B        MOVZWL  #500, -(SP)                                  ;
              00000000G 00                 04  FB 00020        CALLS   #4, NML$CHKEXE                               ;
                        07                 50  E9 00027        BLBC    R0, 1$                                       ;
              00000000G 00                 07  D0 0002A        MOVL    #7, NML$GL_NML_ENTITY                        ; 0841
                   FF72 CF                 6C  FA 00031 1$:    CALLG   (NPARSE_BLOCK), NML$PRSIDN                   ; 0845
                        50                 01  D0 00036        MOVL    #1, R0                                       ; 0846
                                               04 00039        RET                                                  ; 0848

; Routine Size:  58 bytes,    Routine Base:  $CODE$ + 01C6
```

```
1 %SBTTL 'NML$PRS_NODE_NUM_ENTITY  Check node address against executor (action routine)'
1 GLOBAL ROUTINE NML$PRS_NODE_NUM_ENTITY =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This is a NPARSE action that checks the node address against the
1 !       node address of the executor node and then stores it.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NPARSE_BLOCK [NPA$L_FLDPTR] contains the pointer to the entity
1 !           format code and id string.
1 !       NML$GL_PRS_FLGS contains the current message parsing flag information.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NML$GB_ENTITY_FORMAT contains the entity format code.
1 !       NML$AB_ENTITY_ID contains the entity id string.
1 !       NML$GL_NML_ENTITY is set to NML$C_EXECUTOR if this is the executor
1 !       node.
1 !
1 !--
1
2 BEGIN
2
2 $npa_argdef;                          ! Define NPARSE block reference
2
2 BUILTIN
2     CALLG;
2
2 MAP
2     nml$gb_options : BBLOCK [1];
2
2 BIND
2     addr = (.nparse_block [npa$l_fldptr]+1)<0,16> : BBLOCK [2];
2
2 nml$fix_node_num (addr);
2 !
2 ! If the node address in the NICE command matches the executor node
2 ! address then set the flag to indicate it.
2 !
2 IF nml$chkexe (nma$c_pcno_add, .addr, 0, 0) THEN
2     nml$gl_nml_entity = nml$c_executor;
2 !
2 ! Parse the node id normally.
2 !
2 CALLG (.nparse_block, nml$prsid);
2 RETURN nml$_sts_suc
2
1 END;                                  ! End of NML$PRS_NODE_NUM_ENTITY
```

```
                                       0004 00000          .ENTRY  NML$PRS_NODE_NUM_ENTITY, Save R2              ; 0850
              52        14    AC          01  C1 00002          ADDL3   #1, 20(NPARSE_BLOCK), R2                    ; 0888
                                          52  DD 00007          PUSHL   R2                                          ; 0890
                  00000000G   00          01  FB 00009          CALLS   #1, NML$FIX_NODE_NUM
                                          7E  7C 00010          CLRQ    -(SP)                                       ; 0895
                                          62  DD 00012          PUSHL   (R2)
                              7E    01F6  8F  3C 00014          MOVZWL  #502, -(SP)
                  00000000G   00          04  FB 00019          CALLS   #4, NML$CHKEXE
                              07          50  E9 00020          BLBC    R0, 1$
                  00000000G   00          07  D0 00023          MOVL    #7, NML$GL_NML_ENTITY                       ; 0896
                       FEDF   CF          6C  FA 0002A 1$:      CALLG   (NPARSE_BLOCK), NML$PRSID                   ; 0900
                              50          01  D0 0002F          MOVL    #1, R0                                      ; 0901
                                              04 00032          RET                                                 ; 0903

; Routine Size:  51 bytes,    Routine Base:  $CODE$ + 0200
;   918          0904   1 %SBTTL 'NML$PRS_NODE_NUM        Check node address (action routine)'
;   919          0905   1 GLOBAL ROUTINE NML$PRS_NODE_NUM =
;   920          0906   1
;   921          0907   1 !++
;   922          0908   1 ! FUNCTIONAL DESCRIPTION:
;   923          0909   1 !
;   924          0910   1 !       This is a NPARSE action that checks a node address parameter
;   925          0911   1 !       and fixes up the area number (if necessary) and then stores it.
;   926          0912   1 !
;   927          0913   1 ! FORMAL PARAMETERS:
;   928          0914   1 !
;   929          0915   1 !       NONE
;   930          0916   1 !
;   931          0917   1 ! IMPLICIT INPUTS:
;   932          0918   1 !
;   933          0919   1 !       NPARSE_BLOCK [NPA$L_FLDPTR] contains the pointer to the entity
;   934          0920   1 !           format code and id string.
;   935          0921   1 !       NML$GL_PRS_FLGS contains the current message parsing flag information.
;   936          0922   1 !
;   937          0923   1 ! IMPLICIT OUTPUTS:
;   938          0924   1 !
;   939          0925   1 !       NML$GB_ENTITY_FORMAT contains the entity format code.
;   940          0926   1 !       NML$AB_ENTITY_ID contains the entity id string.
;   941          0927   1 !       NML$GL_NML_ENTITY is set to NML$C_EXECUTOR if this is the executor
;   942          0928   1 !       node.
;   943          0929   1 !
;   944          0930   1 !--
;   945          0931   1
;   946          0932   2 BEGIN
;   947          0933   2
;   948          0934   2 $npa_argdef;                      ! Define NPARSE block reference
;   949          0935   2
;   950          0936   2 BUILTIN
;   951          0937   2     CALLG;
;   952          0938   2
;   953          0939   2 BIND
;   954          0940   2     addr = (.nparse_block [npa$l_fldptr]+1)<0,16> : BBLOCK [2];
;   955          0941   2 !
;   956          0942   2 ! Parse the node id normally.
;   957          0943   2 !
;   958          0944   2 nml$fix_node_num (addr);
;   959          0945   2 CALLG (.nparse_block, nml$prsid);
;   960          0946   2 RETURN nml$_sts_suc
;   961          0947   2
;   962          0948   1 END;                              ! End of NML$PRS_NODE_NUM



                                     0000 00000         .ENTRY  NML$PRS_NODE_NUM, Save nothing                 ; 0905
         50      14   AC           01   C1 00002         ADDL3   #1, 20(NPARSE_BLOCK), R0                        ; 0940
                                     50   DD 00007         PUSHL   R0                                              ; 0944
          00000000G   00           01   FB 00009         CALLS   #1, NML$FIX_NODE_NUM                            ;
               FEC6   CF           6C   FA 00010         CALLG   (NPARSE_BLOCK), NML$PRSID                       ; 0945
                      50           01   D0 00015         MOVL    #1, R0                                          ; 0946
                                     04 00018         RET                                                     ; 0948
```

```
; Routine Size:  25 bytes,    Routine Base:  $CODE$ + 0233
```

```
1 %SBTTL 'NML$PRS_MODULE  Check for specified module'
1 GLOBAL ROUTINE NML$PRS_MODULE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine is called during parsing of the module entity id in
1 !       a NICE message.  It's function is to determine the NML internal
1 !       entity code from the module string.  It also saves the module
1 !       id in NML$AB_ENTITY_ID.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
1 !           NPA$L_FLDCNT is the parameter length.
1 !           NPA$L_FLDPTR is a pointer to the parameter in the received
1 !               message buffer.
1 !           NPA$L_PARAM is the module type to check for.
1 !       NML$GL_PRS_FLGS contains the current message parsing flag information.
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NML$GL_NML_ENTITY = the internal NML entity ID of the module.
1 !       NML$AB_ENTITY_ID = the module id string
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       NML$_STS_SUC - the module string corresponds to the one the parsing
1 !               tables currently seek.
1 !       failure - the module string doesn't correspond to the internal
1 !               entity code passed by the parsing tables.
1 !
1 !--
1
2 BEGIN
2
2 $NPA_ARGDEF;
2
2 BUILTIN
2     CALLG;
2
2 LOCAL
2     iptr,
2     length,
2     status;
2
2 status = 0;
2 iptr = .nparse_block [npa$l_fldptr];
2 length = ch$rchar_a (iptr);                   ! Save entity length
2 SELECTONEU .nparse_block [npa$l_param] OF
2     SET
2     [nml$c_x25_access]:
2         status = CH$EQL (.length,
2                          .iptr,
2                          10,
2                          UPLIT (%ASCII 'X25-ACCESS'));
2     [nml$c_protocol]:
2         status = CH$EQL (.length,
; 1021          1006 2                        .iptr,
; 1022          1007 2                        12,
; 1023          1008 2                        UPLIT (%ASCII 'X25-PROTOCOL'));
; 1024          1009 2          [nml$c_x25_serv]:
; 1025          1010 2              status = CH$EQL (.length,
; 1026          1011 2                        .iptr,
; 1027          1012 2                        10,
; 1028          1013 2                        UPLIT (%ASCII 'X25-SERVER'));
; 1029          1014 2          [nml$c_trace]:
; 1030          1015 2              status = CH$EQL (.length,
; 1031          1016 2                        .iptr,
; 1032          1017 2                        9,
; 1033          1018 2                        UPLIT (%ASCII 'X25-TRACE'));
; 1034          1019 2          [nml$c_x29_serv]:
; 1035          1020 2              status = CH$EQL (.length,
; 1036          1021 2                        .iptr,
; 1037          1022 2                        10,
; 1038          1023 2                        UPLIT (%ASCII 'X29-SERVER'));
; 1039          1024 2          [nml$c_ni_config]:
; 1040          1025 3              BEGIN
; 1041          1026 3              status = CH$EQL (.length,
; 1042          1027 3                        .iptr,
; 1043          1028 3                        12,
; 1044          1029 3                        UPLIT (%ASCII 'CONFIGURATOR'));
; 1045          1030 2              END;
; 1046          1031 2          TES;
; 1047          1032 2      !
; 1048          1033 2      ! If the parse tables are checking for the module type in the NICE
; 1049          1034 2      ! message, save the module name.
; 1050          1035 2      !
; 1051          1036 2      IF .status THEN
; 1052          1037 2          CALLG (.nparse_block, nml$prsidn);
; 1053          1038 2      RETURN .status;
; 1054          1039 1      END;                                ! End of NMLSPRS_MODULE


                                                                 .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                  00007          .BLKB   1
      00 00 53 53 45 43 43 41 2D 35 32 58  00008 P.AAB:  .ASCII  \X25-ACCESS\<0><0>                          ;
      4C 4F 43 4F 54 4F 52 50 2D 35 32 58  00014 P.AAC:  .ASCII  \X25-PROTOCOL\                              ;
      00 00 52 45 56 52 45 53 2D 35 32 58  00020 P.AAD:  .ASCII  \X25-SERVER\<0><0>                          ;
      00 00 00 45 43 41 52 54 2D 35 32 58  0002C P.AAE:  .ASCII  \X25-TRACE\<0><0><0>                        ;
      00 00 52 45 56 52 45 53 2D 39 32 58  00038 P.AAF:  .ASCII  \X29-SERVER\<0><0>                          ;
      52 4F 54 41 52 55 47 49 46 4E 4F 43  00044 P.AAG:  .ASCII  \CONFIGURATOR\                              ;


                                                                 .PSECT  $CODE$,NOWRT,2

                                      01FC 00000                 .ENTRY  NMLSPRS_MODULE, Save R2,R3,R4,R5,R6,R7,R8   ; 0950
                   58 00000000'    00   9E 00002                 MOVAB   P.AAB, R8                                   ;
                                   56   D4 00009                 CLRL    STATUS                                      ; 0994
                   57        14    AC   D0 0000B                 MOVL    20(NPARSE_BLOCK), IPTR                      ; 0995
                   55              87   9A 0000F                 MOVZBL  (IPTR)+, LENGTH                             ; 0996
                   50        20    AC   D0 00012                 MOVL    32(NPARSE_BLOCK), R0                        ; 0997
```

```
                                  0D               50  D1 00016       CMPL    R0, #13                                   ; 0999
                                                   0A  12 00019       BNEQ    1$                                        ;
                                                   54  D4 0001B       CLRL    R4                                        ; 1000
        0A            00          67               55  2D 0001D       CMPC5   LENGTH, (IPTR), #0, #10, P.AAB            ;
                                                   68     00022                                                         ;
                                                   2E  11 00023       BRB     4$                                        ;
                                  19               50  D1 00025 1$:   CMPL    R0, #25                                   ; 1004
                                                   0B  12 00028       BNEQ    2$                                        ;
                                                   54  D4 0002A       CLRL    R4                                        ; 1005
        0C            00          67               55  2D 0002C       CMPC5   LENGTH, (IPTR), #0, #12, P.AAC            ;
                                            0C     A8     00031                                                         ;
                                                   1E  11 00033       BRB     4$                                        ;
                                  11               50  D1 00035 2$:   CMPL    R0, #17                                   ; 1009
                                                   0B  12 00038       BNEQ    3$                                        ;
                                                   54  D4 0003A       CLRL    R4                                        ; 1010
        0A            00          67               55  2D 0003C       CMPC5   LENGTH, (IPTR), #0, #10, P.AAD            ;
                                            18     A8     00041                                                         ;
                                                   0E  11 00043       BRB     4$                                        ;
                                  13               50  D1 00045 3$:   CMPL    R0, #19                                   ; 1014
                                                   0D  12 00048       BNEQ    5$                                        ;
                                                   54  D4 0004A       CLRL    R4                                        ; 1015
        09            00          67               55  2D 0004C       CMPC5   LENGTH, (IPTR), #0, #9, P.AAE             ;
                                            24     A8     00051                                                         ;
                                                   22  13 00053 4$:   BEQL    8$                                        ;
                                                   22  11 00055       BRB     9$                                        ;
                                  15               50  D1 00057 5$:   CMPL    R0, #21                                   ; 1019
                                                   0B  12 0005A       BNEQ    6$                                        ;
                                                   54  D4 0005C       CLRL    R4                                        ; 1020
        0A            00          67               55  2D 0005E       CMPC5   LENGTH, (IPTR), #0, #10, P.AAF            ;
                                            30     A8     00063                                                         ;
                                                   0E  11 00065       BRB     7$                                        ;
                                  17               50  D1 00067 6$:   CMPL    R0, #23                                   ; 1024
                                                   10  12 0006A       BNEQ    10$                                       ;
                                                   54  D4 0006C       CLRL    R4                                        ; 1026
        0C            00          67               55  2D 0006E       CMPC5   LENGTH, (IPTR), #0, #12, P.AAG            ;
                                            3C     A8     00073                                                         ;
                                                   02  12 00075 7$:   BNEQ    9$                                        ;
                                                   54  D6 00077 8$:   INCL    R4                                        ;
                                  56               54  D0 00079 9$:   MOVL    R4, STATUS                                ;
                                  05               56  E9 0007C 10$:  BLBC    STATUS, 11$                               ; 1036
                          FE9E    CF               6C  FA 0007F       CALLG   (NPARSE_BLOCK), NMLSPRSIDN                ; 1037
                                  50               56  D0 00084 11$:  MOVL    STATUS, R0                                ; 1038
                                                       04 00087       RET                                               ; 1039

; Routine Size:  136 bytes,    Routine Base:  $CODE$ + 024C
; 1056          1040  1 %SBTTL 'NML$PRS_ACTIVE_NET  Store network format code and id (action routine)'
; 1057          1041  1 GLOBAL ROUTINE NML$PRS_ACTIVE_NET  =
; 1058          1042  1
; 1059          1043  1 !++
; 1060          1044  1 ! FUNCTIONAL DESCRIPTION:
; 1061          1045  1 !       This is a NPARSE action routine that is called when parsing a NICE
; 1062          1046  1 !       command with an X25-Protocol network entity.  It saves a default
; 1063          1047  1 !       network entity of "active network".  This is here in anticipation
; 1064          1048  1 !       of multinetwork support.
; 1065          1049  1 !
; 1066          1050  1 ! IMPLICIT OUTPUTS:
; 1067          1051  1 !
; 1068          1052  1 !       NML$GB_ENTITY_FORMAT contains NMA$C_ENT_ACT (active).
; 1069          1053  1 !--
; 1070          1054  1
; 1071          1055  2 BEGIN
; 1072          1056  2
; 1073          1057  2 !
; 1074          1058  2 ! Use a zero length string to indicate "Active network".
; 1075          1059  2 !
; 1076          1060  2 nml$gb_entity_format = 0;
; 1077          1061  2 nml$ab_entity_id = 0;
; 1078          1062  2
; 1079          1063  2 RETURN nml$_sts_suc
; 1080          1064  2
; 1081          1065  1 END;                                   ! End of NML$PRS_ACTIVE_NET


                                     0000 00000        .ENTRY  NML$PRS_ACTIVE_NET, Save nothing                        ; 1041
                      00000000G  00    94 00002        CLRB    NML$GB_ENTITY_FORMAT                                    ; 1060
                      00000000G  00    D4 00008        CLRL    NML$AB_ENTITY_ID                                        ; 1061
                  50             01    D0 0000E        MOVL    #1, R0                                                  ; 1063
                                       04 00011        RET                                                             ; 1065

; Routine Size:  18 bytes,    Routine Base:  $CODE$ + 02D4
```

```
%SBTTL 'NML$PRSSNKNNA  Parse sink node name'
GLOBAL ROUTINE NML$PRSSNKNNA =

!++
! FUNCTIONAL DESCRIPTION:
!
!       This is a NPARSE action that parses the sink node name.
!       The corresponding address is retrieved and saved for use.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       NPARSE_BLOCK [NPA$L_FLDPTR] contains the address of the node name.
!       NPARSE_BLOCK [NPA$L_FLDCNT] contains the length of the counted node
!               name string (including the count byte).
!       NML$GL_PRS_FLGS contains the current message parsing flag information.
!
! IMPLICIT OUTPUTS:
!
!       NML$GL_PRS_FLGS [NML$V_PRS_EXESNK] is set if this is the executor
!       node.
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       NONE
!
! SIDE EFFECTS:
!
!       NONE
!
!--

BEGIN

$NPA_ARGDEF;                            ! Define NPARSE block reference

MAP
    nml$gb_options      : BBLOCK [1];

LOCAL
    addr : WORD,
    namptr,
    namlen;

!
! Open the node data base file (in case it's a permanent operation).
!
IF .nml$gb_options [nma$v_opt_per] THEN
    nml$openfile (nma$c_opn_node, nma$c_opn_ac_ro);
!
! Save the event sink node address.
!
namptr = .nparse_block [npa$l_fldptr] + 1;
```

```
; 1140          1123  2 namlen = .nparse_block [npa$l_fldcnt] - 1;
; 1141          1124  2
; 1142          1125  2 IF nml$getnodadr (.namlen, .namptr, addr) THEN
; 1143          1126  2     nml$gw_evtsnkadr = .addr
; 1144          1127  2 ELSE
; 1145          1128  2     nml$error_2 (nma$c_sts_ide, nma$c_ent_nod);
; 1146          1129  2 !
; 1147          1130  2 ! If the address matches the executor node address then set the flag
; 1148          1131  2 ! to indicate the executor sink node.
; 1149          1132  2 !
; 1150          1133  2 IF nml$chkexe (nma$c_pcno_add, .addr, 0, 0) THEN
; 1151          1134  2     nml$gl_prs_flgs [nml$v_prs_exesnk] = 1;
; 1152          1135  2 RETURN nml$_sts_suc
; 1153          1136  1 END;                               ! End of NML$PRSSNKNNA



                                                  0000 00000          .ENTRY  NML$PRSSNKNNA, Save nothing                         ; 1067
                               5E               04  C2 00002          SUBL2   #4, SP                                              ;
                                  00000000G     00  95 00005          TSTB    NML$GB_OPTIONS                                      ; 1117
                                                09  18 0000B          BGEQ    1$                                                  ;
                                                7E  7C 0000D          CLRQ    -(SP)                                               ; 1118
                    00000000G   00              02  FB 0000F          CALLS   #2, NML$OPENFILE                                    ;
              51          14    AC              01  C1 00016 1$:      ADDL3   #1, 20(NPARSE_BLOCK), NAMPTR                        ; 1122
              50          10    AC              01  C3 0001B          SUBL3   #1, 16(NPARSE_BLOCK), NAMLEN                        ; 1123
                                         4003   8F  BB 00020          PUSHR   #^M<R0,R1,SP>                                       ; 1125
                    00000000G   00              03  FB 00024          CALLS   #3, NML$GETNODADR                                   ;
                                09              50  E9 0002B          BLBC    R0, 2$                                              ;
                    00000000G   00              6E  B0 0002E          MOVW    ADDR, NML$GW_EVTSNKADR                              ; 1126
                                                0C  11 00035          BRB     3$                                                  ;
                                                7E  D4 00037 2$:      CLRL    -(SP)                                               ; 1128
                               7E               09  CE 00039          MNEGL   #9, -(SP)                                           ;
                    00000000G   00              02  FB 0003C          CALLS   #2, NML$ERROR_2                                     ;
                                                7E  7C 00043 3$:      CLRQ    -(SP)                                               ; 1133
                               7E          08   AE  3C 00045          MOVZWL  ADDR, -(SP)                                         ;
                               7E        01F6   8F  3C 00049          MOVZWL  #502, -(SP)                                         ;
                    00000000G   00              04  FB 0004E          CALLS   #4, NML$CHKEXE                                      ;
                                07              50  E9 00055          BLBC    R0, 4$                                              ;
                    00000000G   00              01  88 00058          BISB2   #1, NML$GL_PRS_FLGS+1                               ; 1134
                               50               01  D0 0005F 4$:      MOVL    #1, R0                                              ; 1135
                                                    04 00062          RET                                                         ; 1136

; Routine Size:  99 bytes,    Routine Base:  $CODE$ + 02E6
```

```
1 %SBTTL 'NML$PRSSNKNAD  Parse sink node address'
1 GLOBAL ROUTINE NML$PRSSNKNAD =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This is a NPARSE action routine that stores the sink node address.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NPARSE_BLOCK [NPA$L_FLDPTR] points to the node address.
1 !       NPARSE_BLOCK [NPA$L_FLDCNT] contains the count of the address plus
1 !               the NMA$C_ENT_ADD byte.
1 !       NML$GL_PRS_FLGS contains the current message parsing flag information.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NML$GL_PRS_FLGS [NML$V_PRS_EXESNK] is set if this is the executor node.
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2 BEGIN
2
2 $NPA_ARGDEF;                          ! Define NPARSE block reference
2
2 MAP
2     nml$gb_options      : BBLOCK [1];
2
2 BUILTIN
2     CALLG;
2
2 BIND
2     addr = (.nparse_block [npa$l_fldptr]+1)<0,16>;
2
2 !
2 ! Open the node data base file (in case it's a permanent operation.
2 !
2 IF .nml$gb_options [nma$v_opt_per] THEN
2     nml$openfile (nma$c_opn_node, nma$c_opn_ac_ro);
2 !
2 ! If the address is zero then get the real executor node address and
2 ! set the flag indicating the executor sink node.
2 !
2 IF .addr EQLU 0 THEN
```

```
; 1212          1194    3           BEGIN
; 1213          1195    3           nml$getexeadr (addr);
; 1214          1196    3           nml$gl_prs_flgs [nml$v_prs_exesnk] = 1;
; 1215          1197    3           END
; 1216          1198    2       ELSE
; 1217          1199    3           BEGIN
; 1218          1200    3           !
; 1219          1201    3           ! If the node address has an area number of 0, fix it up to something
; 1220          1202    3           ! meaningful.
; 1221          1203    3           !
; 1222          1204    3           nml$fix_node_num (addr);
; 1223          1205    3           !
; 1224          1206    3           ! If the address matches the executor node address then set the flag
; 1225          1207    3           ! to indicate the executor sink node.
; 1226          1208    3           !
; 1227          1209    3           IF nml$chkexe (nma$c_pcno_add, .addr, 0, 0) THEN
; 1228          1210    3               nml$gl_prs_flgs [nml$v_prs_exesnk] = 1;
; 1229          1211    2           END;
; 1230          1212    2
; 1231          1213    2       nml$gw_evtsnkadr = .addr;
; 1232          1214    2       RETURN nml$_sts_suc
; 1233          1215    1       END;                                    ! End of NML$PRSSNKNAD


                                                  0004 00000          .ENTRY  NML$PRSSNKNAD, Save R2                              ; 1138
                    52        14   AC         01   C1 00002           ADDL3   #1, 20(NPARSE_BLOCK), R2                            ; 1182
                                      00000000G 00   95 00007         TSTB    NML$GB_OPTIONS                                      ; 1187
                                              09   18 0000D           BGEQ    1$                                                  ;
                                              7E   7C 0000F           CLRQ    -(SP)                                               ; 1188
                       00000000G  00          02   FB 00011           CALLS   #2, NML$OPENFILE                                    ;
                                              62   D5 00018 1$:       TSTL    (R2)                                                ; 1193
                                              0B   12 0001A           BNEQ    2$                                                  ;
                                              52   DD 0001C           PUSHL   R2                                                  ; 1195
                       00000000G  00          01   FB 0001E           CALLS   #1, NML$GETEXEADR                                   ;
                                              1C   11 00025           BRB     3$                                                  ; 1196
                                              52   DD 00027 2$:       PUSHL   R2                                                  ; 1204
                       00000000G  00          01   FB 00029           CALLS   #1, NML$FIX_NODE_NUM                                ;
                                              7E   7C 00030           CLRQ    -(SP)                                               ; 1209
                                              62   DD 00032           PUSHL   (R2)                                                ;
                                  7E   01F6   8F   3C 00034           MOVZWL  #502, -(SP)                                         ;
                       00000000G  00          04   FB 00039           CALLS   #4, NML$CHKEXE                                      ;
                                  07          50   E9 00040           BLBC    R0, 4$                                              ;
                       00000000G  00          01   88 00043 3$:       BISB2   #1, NML$GL_PRS_FLGS+1                               ; 1210
                       00000000G  00          62   B0 0004A 4$:       MOVW    (R2), NML$GW_EVTSNKADR                              ; 1213
                                  50          01   D0 00051           MOVL    #1, R0                                              ; 1214
                                                   04 00054           RET                                                         ; 1215

; Routine Size:  85 bytes,    Routine Base:  $CODE$ + 0349
```

```
1 %SBTTL 'NML$PRSEXESNK  Get event sink executor node address'
1 GLOBAL ROUTINE NML$PRSEXESNK =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine is called while parsing a NICE message logging entity.
1 !       It sets up the default sink node as the executor node if no sink
1 !       node was specified explicitly.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
1 !           NPA$L_FLDCNT is the parameter length.
1 !           NPA$L_FLDPTR is a pointer to the parameter in the received
1 !               message buffer.
1 !       NML$GL_PRS_FLGS contains the current message parsing flag information.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NML$GL_PRS_FLGS [NML$V_PRS_SNKNOD] is set if it was not previously
1 !           set.
1 !       NML$GL_PRS_FLGS [NML$V_PRS_EXESNK] is set if the executor node
1 !           address was found in the data base and a sink node had not been
1 !           previously specified.
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       Always returns success (NML$_STS_SUC).
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2 BEGIN
2
2 $NPA_ARGDEF;
2
2 MAP
2     nml$gb_options      : BBLOCK [1];
2
2 LOCAL
2     addr : WORD;
2 !
2 ! If no sink node has been specified then the executor node is intended.
2 !
2 IF NOT .nml$gl_prs_flgs [nml$v_prs_snknod] THEN
3     BEGIN
3     !
3     ! Open node file if it's a permanent data base operation.
```

```
                                                                              B  7
NML$PARINI          NML initial message parsing module               16-Sep-1984 00:23:43     VAX-11 Bliss-32 V4.0-742                Page  38
V04-000             NML$PRSEXESNK  Get event sink executor node add  14-Sep-1984 12:50:15     [NML.SRC]NMLPARINI.B32;1                  (20)

; 1292              1273  3         !
; 1293              1274  3         IF .nml$gb_options [nma$v_opt_per] THEN
; 1294              1275  3             nml$openfile (nma$c_opn_node, nma$c_opn_ac_ro);
; 1295              1276  3         !
; 1296              1277  3         ! Get the executor node address.  If none is specified, use address 0.
; 1297              1278  3         !
; 1298              1279  3         IF nml$getexeadr (addr) THEN
; 1299              1280  3             nml$gw_evtsnkadr = .addr
; 1300              1281  3         ELSE
; 1301              1282  3             nml$gw_evtsnkadr = 0;
; 1302              1283  3         nml$gl_prs_flgs [nml$v_prs_snknod] = 1;
; 1303              1284  3         nml$gl_prs_flgs [nml$v_prs_exesnk] = 1;
; 1304              1285  2         END;
; 1305              1286  2     RETURN nml$_sts_suc
; 1306              1287  2
; 1307              1288  1     END;                                  ! End of NML$PRSEXESNK




                                                     0004 00000            .ENTRY  NML$PRSEXESNK, Save R2                             ; 1217
                        52 00000000G  00   9E 00002            MOVAB   NML$GW_EVTSNKADR, R2                                       ;
                        5E            04   C2 00009            SUBL2   #4, SP                                                     ;
          2B 00000000G  00            01   E0 0000C            BBS     #1, NML$GL_PRS_FLGS+1, 4$                                  ; 1269
                           00000000G  00   95 00014            TSTB    NML$GB_OPTIONS                                             ; 1274
                                      09   18 0001A            BGEQ    1$                                                         ;
                                      7E   7C 0001C            CLRQ    -(SP)                                                      ; 1275
             00000000G  00            02   FB 0001E            CALLS   #2, NML$OPENFILE                                           ;
                                      5E   DD 00025 1$:        PUSHL   SP                                                         ; 1279
             00000000G  00            01   FB 00027            CALLS   #1, NML$GETEXEADR                                          ;
                        05            50   E9 0002E            BLBC    R0, 2$                                                     ;
                        62            6E   B0 00031            MOVW    ADDR, NML$GW_EVTSNKADR                                     ; 1280
                                      02   11 00034            BRB     3$                                                         ;
                                      62   B4 00036 2$:        CLRW    NML$GW_EVTSNKADR                                           ; 1282
             00000000G  00            03   88 00038 3$:        BISB2   #3, NML$GL_PRS_FLGS+1                                      ; 1284
                        50            01   D0 0003F 4$:        MOVL    #1, R0                                                     ; 1286
                                           04 00042            RET                                                                ; 1288

; Routine Size:  67 bytes,    Routine Base:  $CODE$ + 039E
; 1309            1289  1 %SBTTL 'NML$PRSDEVICE  Check device id (action routine)'
; 1310            1290  1 GLOBAL ROUTINE NML$PRSDEVICE =
; 1311            1291  1
; 1312            1292  1 !++
; 1313            1293  1 ! FUNCTIONAL DESCRIPTION:
; 1314            1294  1 !       This is an NPARSE action that saves line and circuit IDs.  This
; 1315            1295  1 !       a separate routine so that wildcarding can be added later.
; 1316            1296  1 !
; 1317            1297  1 ! IMPLICIT INPUTS:
; 1318            1298  1 !       NPARSE_BLOCK [NPA$L_FLDPTR] contains the pointer to the entity
; 1319            1299  1 !            format code and id string.
; 1320            1300  1 !
; 1321            1301  1 ! IMPLICIT OUTPUTS:
; 1322            1302  1 !       NML$GB_ENTITY_FORMAT contains the entity format code.
; 1323            1303  1 !       NML$AB_ENTITY_ID contains the entity id string.
; 1324            1304  1 !
; 1325            1305  1 !--
; 1326            1306  1
; 1327            1307  2 BEGIN
; 1328            1308  2
; 1329            1309  2 $NPA_ARGDEF;                       ! Define NPARSE block reference
; 1330            1310  2
; 1331            1311  2 BUILTIN
; 1332            1312  2         CALLG;
; 1333            1313  2
; 1334            1314  2 LOCAL
; 1335            1315  2         length,
; 1336            1316  2         addr;
; 1337            1317  2
; 1338            1318  2 length = .nparse_block [npa$l_fldcnt] - 1; ! Get length not including count
; 1339            1319  2 addr = .nparse_block [npa$l_fldptr] + 1;   ! Get address of byte after count
; 1340            1320  2
; 1341            1321  2 !****************************************************
; 1342            1322  2 !* Wild cards are not currently allowed in line
; 1343            1323  2 !* specifications.
; 1344            1324  2
; 1345            1325  2 IF CH$FIND_CH (.length, .addr, %C'*') THEN
; 1346            1326  3         BEGIN
; 1347            1327  3 !       nml$gl_prs_flgs = .nml$gl_prs_flgs AND lin$m_wildcards;
; 1348            1328  3         RETURN nml$_sts_ide;
; 1349            1329  2         END;
; 1350            1330  2
; 1351            1331  2 !*
; 1352            1332  2 !*
; 1353            1333  2 !****************************************************
; 1354            1334  2
; 1355            1335  2 CALLG (.nparse_block, nml$prsidn); ! Save line entity id and format
; 1356            1336  2 RETURN nml$_sts_suc;
; 1357            1337  1 END;                               ! End of NML$PRSDEVICE


                                         0000 00000         .ENTRY  NML$PRSDEVICE, Save nothing                        ; 1290
                51       10   AC           01  C3 00002         SUBL3   #1, 16(NPARSE_BLOCK), LENGTH                       ; 1318
                50       14   AC           01  C1 00007         ADDL3   #1, 20(NPARSE_BLOCK), ADDR                         ; 1319
```

```
                  60          51            2A  3A 0000C          LOCC     #42, LENGTH, (ADDR)                            ; 1325
                                            02  12 00010          BNEQ     1$                                             ;
                                            51  D4 00012          CLRL     R1                                             ;
                              04            51  E9 00014 1$:      BLBC     R1, 2$                                         ;
                              50            12  CE 00017          MNEGL    #18, R0                                        ; 1328
                                                04 0001A          RET                                                     ;
                      FD6D    CF            6C  FA 0001B 2$:      CALLG    (NPARSE_BLOCK), NML$PRSIDN                     ; 1335
                              50            01  D0 00020          MOVL     #1, R0                                         ; 1336
                                                04 00023          RET                                                     ; 1337

; Routine Size:  36 bytes,    Routine Base:  $CODE$ + 03E1
; 1359          1338  1 %SBTTL 'NML$PRSLOGSIN  Logging sink node check (action routine)'
; 1360          1339  1 GLOBAL ROUTINE NML$PRSLOGSIN =
; 1361          1340  1
; 1362          1341  1 !++
; 1363          1342  1 ! FUNCTIONAL DESCRIPTION:
; 1364          1343  1 !
; 1365          1344  1 !       This is a NPARSE action routine that checks the function code
; 1366          1345  1 !       for a read function.  If the function is read then failure is
; 1367          1346  1 !       returned to indicate that a sink node id must be parsed.
; 1368          1347  1 !       If function is not read then success is returned.
; 1369          1348  1 !
; 1370          1349  1 ! FORMAL PARAMETERS:
; 1371          1350  1 !
; 1372          1351  1 !       NONE
; 1373          1352  1 !
; 1374          1353  1 ! IMPLICIT INPUTS:
; 1375          1354  1 !
; 1376          1355  1 !       NML$GB_FUNCTION contains the function code.
; 1377          1356  1 !
; 1378          1357  1 ! IMPLICIT OUTPUTS:
; 1379          1358  1 !
; 1380          1359  1 !       NONE
; 1381          1360  1 !
; 1382          1361  1 ! ROUTINE VALUE:
; 1383          1362  1 ! COMPLETION CODES:
; 1384          1363  1 !
; 1385          1364  1 !       Success (NML$_STS_SUC) is returned if the funtion is not read.
; 1386          1365  1 !       Otherwise, failure (NML$_STS_MPR) is indicated.
; 1387          1366  1 !
; 1388          1367  1 ! SIDE EFFECTS:
; 1389          1368  1 !
; 1390          1369  1 !       NONE
; 1391          1370  1 !
; 1392          1371  1 !--
; 1393          1372  1
; 1394          1373  2 BEGIN
; 1395          1374  2
; 1396          1375  2 $NPA_ARGDEF;                          ! Define NPARSE block reference
; 1397          1376  2
; 1398          1377  2 IF .nml$gb_function NEQU nma$c_fnc_rea THEN
; 1399          1378  2     RETURN nml$_sts_suc
; 1400          1379  2 ELSE
; 1401          1380  2     RETURN nml$_sts_mpr;
; 1402          1381  2
; 1403          1382  1 END;                                  ! End of NML$PRSLOGSIN


                                 0000 00000          .ENTRY  NML$PRSLOGSIN, Save nothing               ; 1339
          14 00000000G   00   91 00002          CMPB    NML$GB_FUNCTION, #20                      ; 1377
                         04   13 00009          BEQL    1$                                        ;
          50             01   D0 0000B          MOVL    #1, R0                                    ; 1380
                              04 0000E          RET                                               ;
          50             0A   CE 0000F 1$:      MNEGL   #10, R0                                   ;
                              04 00012          RET                                               ; 1382
```

```
; Routine Size:  19 bytes,    Routine Base:  $CODE$ + 0405
; 1405          1383  1 %SBTTL 'NML$PRS_NOREAD Check function code (action routine)'
; 1406          1384  1 GLOBAL ROUTINE NML$PRS_NOREAD =
; 1407          1385  1
; 1408          1386  1 !++
; 1409          1387  1 ! FUNCTIONAL DESCRIPTION:
; 1410          1388  1 !       Check the saved function code and return success if it's
; 1411          1389  1 !       not "read"
; 1412          1390  1 !
; 1413          1391  1 ! ROUTINE VALUE:
; 1414          1392  1 ! COMPLETION CODES:
; 1415          1393  1 !       Returns success (NML$_STS_SUC) if the function code is "read".
; 1416          1394  1 !       Otherwise it returns NML$_STS_CMP.
; 1417          1395  1 !
; 1418          1396  1 !--
; 1419          1397  1
; 1420          1398  2 BEGIN
; 1421          1399  2
; 1422          1400  2 $NPA_ARGDEF;                          ! Define NPARSE block reference
; 1423          1401  2
; 1424          1402  2 IF .nml$gb_function EQL nma$c_fnc_rea THEN
; 1425          1403  2     RETURN nml$_sts_cmp
; 1426          1404  2 ELSE
; 1427          1405  2     RETURN nml$_sts_suc;
; 1428          1406  1 END;                                  ! End of NML$PRS_NOREAD



                                          0000 00000          .ENTRY  NML$PRS_NOREAD, Save nothing                          ; 1384
                         14 00000000G  00   91 00002          CMPB    NML$GB_FUNCTION, #20                                  ; 1402
                                       04   12 00009          BNEQ    1$                                                    ;
                         50            10   CE 0000B          MNEGL   #16, R0                                               ; 1405
                                            04 0000E          RET                                                           ;
                         50            01   D0 0000F 1$:      MOVL    #1, R0                                                ;
                                            04 00012          RET                                                           ; 1406

; Routine Size:  19 bytes,    Routine Base:  $CODE$ + 0418
; 1430          1407  1 %SBTTL 'NML$PRSERR1  Error parsing message (action routine)'
; 1431          1408  1 GLOBAL ROUTINE NML$PRSERR1 =
; 1432          1409  1
; 1433          1410  1 !++
; 1434          1411  1 ! FUNCTIONAL DESCRIPTION:
; 1435          1412  1 !
; 1436          1413  1 !     This routine causes an error message to be signalled with the status
; 1437          1414  1 !     code specified in the NPARSE block (NPA$L_PARAM).
; 1438          1415  1 !
; 1439          1416  1 ! FORMAL PARAMETERS:
; 1440          1417  1 !
; 1441          1418  1 !         NONE
; 1442          1419  1 !
; 1443          1420  1 ! IMPLICIT INPUTS:
; 1444          1421  1 !
; 1445          1422  1 !         NONE
; 1446          1423  1 !
; 1447          1424  1 ! IMPLICIT OUTPUTS:
; 1448          1425  1 !
; 1449          1426  1 !         NONE
; 1450          1427  1 !
; 1451          1428  1 ! ROUTINE VALUE:
; 1452          1429  1 ! COMPLETION CODES:
; 1453          1430  1 !
; 1454          1431  1 !         Always returns success (NML$_STS_SUC).
; 1455          1432  1 !
; 1456          1433  1 ! SIDE EFFECTS:
; 1457          1434  1 !
; 1458          1435  1 !         An error message is signalled.
; 1459          1436  1 !
; 1460          1437  1 !--
; 1461          1438  1
; 1462          1439  2 BEGIN
; 1463          1440  2
; 1464          1441  2 $NPA_ARGDEF;                       ! Define NPARSE block reference
; 1465          1442  2
; 1466          1443  2 nml$error_1 (.nparse_block [npa$l_param]); ! Signal message
; 1467          1444  2
; 1468          1445  2 RETURN nml$_sts_suc
; 1469          1446  2
; 1470          1447  1 END;                               ! End of NML$PRSERR1


                                        0000 00000          .ENTRY  NML$PRSERR1, Save nothing                     ; 1408
                               20    AC   DD 00002          PUSHL   32(NPARSE_BLOCK)                              ; 1443
              00000000G  00          01   FB 00005          CALLS   #1, NML$ERROR_1                               ;
                         50          01   D0 0000C          MOVL    #1, R0                                        ; 1445
                                          04 0000F          RET                                                   ; 1447

; Routine Size:  16 bytes,    Routine Base:  $CODE$ + 042B
; 1472          1448  1 %SBTTL 'NML$PRSIDERR  Error parsing entity id (action routine)'
; 1473          1449  1 GLOBAL ROUTINE NML$PRSIDERR =
; 1474          1450  1
; 1475          1451  1 !++
; 1476          1452  1 ! FUNCTIONAL DESCRIPTION:
; 1477          1453  1 !
; 1478          1454  1 !     This routine causes an entity id error message to be signalled
; 1479          1455  1 !     with the detail code specified in the NPARSE block (NPA$L_PARAM).
; 1480          1456  1 !
; 1481          1457  1 ! FORMAL PARAMETERS:
; 1482          1458  1 !
; 1483          1459  1 !         NONE
; 1484          1460  1 !
; 1485          1461  1 ! IMPLICIT INPUTS:
; 1486          1462  1 !
; 1487          1463  1 !         NONE
; 1488          1464  1 !
; 1489          1465  1 ! IMPLICIT OUTPUTS:
; 1490          1466  1 !
; 1491          1467  1 !         NONE
; 1492          1468  1 !
; 1493          1469  1 ! ROUTINE VALUE:
; 1494          1470  1 ! COMPLETION CODES:
; 1495          1471  1 !
; 1496          1472  1 !         Always returns success (NML$_STS_SUC).
; 1497          1473  1 !
; 1498          1474  1 ! SIDE EFFECTS:
; 1499          1475  1 !
; 1500          1476  1 !         NONE
; 1501          1477  1 !
; 1502          1478  1 !--
; 1503          1479  1
; 1504          1480  2 BEGIN
; 1505          1481  2
; 1506          1482  2 $NPA_ARGDEF;                          ! Define NPARSE block reference
; 1507          1483  2
; 1508          1484  2 nml$error_2 (nma$c_sts_ide,
; 1509          1485  2             .nparse_block [npa$l_param]); ! Signal message
; 1510          1486  2
; 1511          1487  2 RETURN nml$_sts_suc
; 1512          1488  2
; 1513          1489  1 END;                                  ! End of NML$PRSERR1



                                        0000 00000        .ENTRY  NML$PRSIDERR, Save nothing                  : 1449
                                 20   AC  DD 00002        PUSHL   32(NPARSE_BLOCK)                            : 1485
                       7E             09  CE 00005        MNEGL   #9, -(SP)                                   : 1484
              00000000G 00            02  FB 00008        CALLS   #2, NML$ERROR_2                             :
                       50             01  D0 0000F        MOVL    #1, R0                                      : 1487
                                          04 00012        RET                                                 : 1489

; Routine Size:  19 bytes,    Routine Base:  $CODE$ + 043B
; 1515          1490  1 END                                     ! End of module
; 1516          1491  1
; 1517          1492  0 ELUDOM



;                                       PSECT SUMMARY
;
;       Name                        Bytes                       Attributes
;
;  $PLIT$                               80  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                             1102  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                               Library Statistics
;
;                                               -------- Symbols --------       Pages       Processing
;       File                                    Total   Loaded  Percent         Mapped      Time
;
;  _$255$DUA28:[NML.OBJ]NMLLIB.L32;1              341       41       12            27         00:00.1
;  _$255$DUA28:[SHRLIB]NMALIBRY.L32;1             887       14        1            47         00:00.2
;  _$255$DUA28:[SYSLIB]STARLET.L32;1             9776        2        0           581         00:02.2



;                                       COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:NMLPARINI/OBJ=OBJ$:NMLPARINI MSRC$:NMLPARINI/UPDATE=(ENH$:NMLPARINI)

; Size:          1102 code + 80 data bytes
; Run Time:         00:25.5
; Elapsed Time:     01:03.5
; Lines/CPU Min:     3517
; Lexemes/CPU-Min: 11092
; Memory Used:  111 pages
; Compilation Complete
```

0285 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY